REMARKS

Applicants have studied the Office Action dated May 2, 2008, and have made amendments to the claims. Applicants respectfully request entry of this amendment under the provisions of 37 C.F.R. § 1.116(a) in that it places the application and claims in condition for allowance or, at least, presents the application in better form for appeal. It is submitted that the application, as amended, is in condition for allowance. By virtue of this amendment, claims 1, 2, 4-8, and 21-24 are pending. Claims 9, 10, and 13-20 have been canceled without prejudice. Claims 1, 6-8, and 22 have been amended. Reconsideration and allowance of the pending claims in view of the above amendments and the following remarks are respectfully requested.

As an initial matter, Applicants submit that the claim amendments made herein do not raise new issues in the application. Claims 1, 6-8, and 22 have been amended to clarify previously-presented claim language. None of these changes raises new issues in the application. Applicants submit that the present amendment places the application in condition for allowance or, at least, presents the application in better form for appeal. Entry of the present amendment is therefore respectfully requested.

Claims 1, 9, 17, and 24 were rejected under 35 U.S.C. § 112, first paragraph, as failing to comply with the written description requirement. The Examiner stated that there was no antecedent basis for the recited features of "directly registering", "directly notifying", and "nested configuration values". Claims 9 and 17 have been canceled so, with respect to these claims, this rejection is moot. With respect to claims 1 and 24, this rejection is respectfully traversed.

With respect to a written description of the features of directly registering and directly notifying, the Examiner is directed to paragraphs 21, 23, 43, 58, and 65 of the specification as originally filed. These paragraphs explain that an application component can register itself in a particular node in a tree, and therefore the HCM notifies the registered component when a modification to a configuration value occurs. While the phrases "registering ... directly" and

"directly notifying" themselves are not in these exact words used in the specification, this feature of the present invention is clearly explained in these paragraphs. From the description in these paragraphs, and the specification as a whole, one of ordinary skill in the art would have understood that the recited features of directly registering and directly notifying were being performed. Because the specification contains a description of the claimed invention, even if not in the identical words used in the claims, the Examiner must provide reasons why one of ordinary skill in the art would not consider the description sufficient. See In re Alton (Fed. Cir. 1996).[1]

With respect to a written description of the feature of "the at least one configuration value and the another configuration value are nested under a common sub-tree in the tree," the Examiner is directed to paragraph 42 of the specification as originally filed. This paragraph states that "configuration data may overlap, the HCM nests them under a common sub-tree." Thus, the recited feature of "the at least one configuration value and the another configuration value are nested under a common sub-tree in the tree" is also supported by the specification as originally filed.

Accordingly, it is respectfully submitted that the rejection of claims 1 and 24 under 35 U.S.C. § 112, first paragraph, should be withdrawn.

Claims 1, 2, 4-10 and 13-24 were rejected under 35 U.S.C. § 102(b) as being anticipated by Hutsch et al. (U.S. Patent Application Publication No. 2001/0034771). Claims 9, 10, and 13-20 have been canceled so, with respect to these claims, this rejection is moot. With respect to claims 1, 2, 4-8, and 21-24, this rejection is respectfully traversed.

The present invention is directed to an efficient and easy-to-implement method for managing configuration data. One embodiment of the present invention provides a method for managing configuration data. According to the method, configuration values are stored in a hierarchical tree having multiple nodes, a defined structure, and defined data types for the stored configuration values.

---

1 See also Fujikawa v. Wattanasin (Fed. Cir. 1996); In re Edwards, 568 F.2d 1349, 1351, 196 USPQ 465, 467 (CCPA 1978) (Ipsis verbis disclosure is not necessary to satisfy the written description requirement of section 112. Instead, the disclosure need only reasonably convey to persons skilled in the art that the inventor had possession of the subject matter in question).

At least one configuration value is stored in each node of these nodes, and each of the configuration values dictates how an application component associated with that configuration value behaves and/or interacts with other application components. Additionally, some of the nodes only store a set of configuration values while other of the nodes store a combination of a set of configuration values and an identifier associated with at least one application component. An application component is directly registered with at least one of the nodes of the tree, based on a query received from the application component. The at least one node comprises at least one configuration value that dictates how the application component behaves with and/or interacts with other application components. The application component is directly notified when a configuration value stored in the at least one node of the tree is modified, based on an addition or change in at least one configuration value that matches the query.

For example, in the exemplary embodiment disclosed in the specification, configuration data, which includes configuration values, is organized into a hierarchical tree. Application components register interest in a particular node of the tree, or a particular sub-tree. The registered interest is stored within the node. Because they registered, these components receive notifications whenever a corresponding configuration value changes. Thus, any component that is a current holder of a registration or reference will be notified when a configuration value changes. Because the configuration values are arranged hierarchically in a tree such as an XML tree, the various application components can register for callbacks or notification upon modification of any nodes in any sub-tree.

Further, the structure of the configuration data in the XML tree can be altered, for example by adding more sub-nodes to a particular node in the tree. An interested party to the changed sub-tree will receive the new configuration data. Thus, an application component can register itself as an interested party to any change in a configuration value. Also, the use of XML to represent the tree allows a sub-tree of configuration data to be easily expanded, with no change in how an application component handles those configuration values. Thus, an application component (such as a Graphical User Interface, an object, an Application Programming Interface, a plug-in, an application, or a user)

has the ability to handle changing configuration data, and can handle configuration data in a hierarchical structure such as by using the extensibility of XML.

The Hutsch reference is directed toward a network portal system that allows universal and integral use of different services by arbitrary client systems. The network portal system links, via a communication network, multiple content provider systems with multiple client systems. The network portal system allows a client to get content from a provider system even if the client system and the provider system do not use the same communication protocol. In Hutsch, a web-top manager within the portal system receives a content request from a client system. The web-top manager communicates with a universal content broker system that is also in the portal system. Upon receipt of the content request from the web-top manager, the universal content broker selects a content provider system that is able to provide the requested content. The universal content broker accesses the selected content provider system, and issues a request that results in the performance of the requested action by the selected content provider system.

In the system of Hutsch, if the request was to retrieve content, the content in a raw data format is passed to the web-top manager. The web-top manager renders the requested content into a page that can be displayed by the requesting client system, and then this page is returned to the requesting client system. The universal content broker system of Hutsch can include a configuration server and a configuration proxy that is coupled to the configuration server. The configuration server includes a first DOM tree, which in turn includes user profiles and application profiles. The configuration proxy includes a second DOM tree, which in turn includes a subset of the data in the first DOM tree.

Amended claim 1 recites:

> storing a plurality of configuration values in a hierarchical tree having a plurality of nodes, a defined structure, and defined data types for the stored configuration values, wherein the plurality of nodes includes at least one inner node and at least one child node that is associated with the inner node, wherein <u>at least one configuration value is stored in each node of the plurality of nodes</u>, and each of the configuration values dictates how an application

> component associated with that configuration value at least one of behaves and interacts with other application components, and wherein some of the nodes only store a set of configuration values while other of the nodes store a combination of a set of configuration values and an identifier associated with at least one application component;
>
> registering at least one application component directly with at least one of the nodes of the tree, based on at least one query received from the at least one application component, wherein the at least one node comprises at least one configuration value that dictates how the application component at least one of behaves with and interacts with other application components; and
>
> directly notifying the at least one application component when a configuration value stored in the at least one node is modified, based on an addition or change in at least one configuration value that matches the at least one query.

Hutsch explicitly teaches that data only exists at leaf nodes of a tree. In contrast, in embodiments of the present invention configuration values exist at all nodes, not just leaf nodes. With respect to this, the Examiner has taken the position that:

> The claim language states "wherein each node is associated with at least one of the configuration values." When given the broadest reasonable interpretation, the concept of associating each node with a configuration value is not considered to restrict the interpretation of the claim limitation to mean that configuration values exist at all nodes. The configuration values of Hutsch are stored in a tree. The tree consists of parent-child relationships. Therefore, even if the values are only stored in the leaf nodes, the nodes located above the leaf node in the hierarchy have an association with the values in the leaf node.
>
> Furthermore, page 8, lines 14-16 of the Applicants' specification states that "In one embodiment of the present invention, each node of the tree includes at least one configuration value." Therefore, since this is just one embodiment, the specification does not explicitly limit the term "associated" to mean that "a configuration value exists at all nodes."

Applicants have amended claim 1 to more clearly recite "wherein at least one configuration value is stored in each node of the plurality of nodes". In contrast, Hutsch teaches that data **only** exists at leaf nodes of a tree. Amended claim 1, on the other hand, now clearly recites that

configuration values exist at all nodes and not just leaf nodes. Accordingly, the claimed invention distinguishes over Hutsch.

Additionally, in embodiments of the present invention the application directly registers with the node itself. Stated differently, the reference to an application component that is to be notified when a change occurs is stored within the node itself. Hutsch teaches that listeners can register to be notified when alterations are made to configuration service elements. These listeners then notify the applications affected. Because data is only stored at leaf nodes, the listeners or components can only register an interest for data in a leaf node. With respect to this, the Examiner stated:

> The examiner respectfully disagrees that the concept of an application component registering as a listener fails to meet this limitation when given the broadest reasonable interpretation of the claimed limitation. The claim limitation is not considered to be explicitly limited to "storing a reference in the node itself' and therefore the Applicants' interpretation of the claim language is not considered to be the only possible interpretation.

Claim 1 has been amended to more clearly recite "registering at least one application component directly with at least one of the nodes of the tree, based on at least one query received from the at least one application component, wherein the at least one node comprises at least one configuration value that dictates how the application component at least one of behaves with and interacts with other application components". Hutsch actually teaches away from this feature of an application component directly registering with a node. Hutsch only teaches that a "listener" registers with a leaf node and that this listener is notified of an alteration made to a configuration service element. The "listener" then notifies the applications that are affected. The Examiner seems to equate the "listener" of Hutsch to the recited "application component". However, nowhere does Hutsch teach or suggest that the information at a leaf node dictates how the application component that is directly registered at a node behaves with and/or interacts with other application components. Stated differently, for the "listener" of Hutsch to equate with the recited "application component", Hutsch would have to teach that the configuration service elements are the same as configuration values and that the configuration service elements dictate how the "listener" interacts with and/or

behaves with other application components. Hutsch clearly does not contain such a teaching. Accordingly, the claimed invention distinguishes over Hutsch.

In embodiments of the present invention, some of the nodes are only associated with a set of configuration values, and other of the nodes are associated with a combination of a set of configuration values and an identifier associated with at least one application component. With respect to this, the Examiner stated that:

> The examiner respectfully disagrees. Paragraph [0159] of Hutsch that subtrees exist. Subtrees are considered to be associated with a plurality of configuration values and identifiers.

Amended claim 1 now clearly recites that each node **stores** at least one configuration value. Claim 1, as amended, also recites "some of the nodes only **store** a set of configuration values while other of the nodes **store** a combination of a set of configuration values and an identifier associated with at least one application component". Hutsch explicitly teaches throughout the disclosure that only leaf nodes of a tree include data. For example, paragraph 329 of Hutsch states that: "in this embodiment, only the leaves of DOM tree 1570 can hold data. The inner nodes are used to show hierarchical relationships." Hutsch does not teach or suggest that each node **stores** at least one configuration value, and some of the nodes only **store** a set of configuration values while other of the nodes **store** a combination of a set of configuration values and an identifier associated with at least one application component". Accordingly, the claimed invention distinguishes over Hutsch.

Furthermore, Hutsch does not teach or suggest "directly notifying the at least one application component when a configuration value stored in the at least one node is modified, based on an addition or change in at least one configuration value that matches the at least one query." Hutsch merely teaches notifying a "listener" that was registered, and the "listener" is not an application component in which the configuration value dictates how the component interacts with and/or behaves with other application components. Accordingly, the claimed invention distinguishes over Hutsch.

A rejection under 35 U.S.C. § 102(b) requires that a single reference teach (i.e., identically describe) each and every element of the rejected claims. Hutsch does not teach each and every element recited in amended claim 1.

Applicants believe that the differences between Hutsch and the present invention are clear in amended claim 1, which sets forth a method for managing configuration data according to an embodiment of the present invention. Therefore, claim 1 distinguishes over the Hutsch reference, and the rejection of this claims under 35 U.S.C. § 102(b) should be withdrawn.

As discussed above, amended claim 1 distinguishes over the Hutsch reference, and thus, claims 2, 4-8, and 21-24 (which depend from claim 1) also distinguish over the Hutsch reference. Further, it is submitted that limitations recited in the dependent claims are not taught by Hutsch. For example, with respect to claim 21, the Examiner states that Hutsch teaches "the plurality of configuration values in the hierarchical tree includes all of the configuration data values that are required by the at least one application component (see [0158]) and [0159])". However, paragraphs 158 and 159 of Hutsch merely state:

> All of the entries are stored in hierarchical form in key-value pairs. As explained more completely, below a configuration tree contains separate branches for this purpose, which can be used for different users or the various user devices. The configuration tree is described through a Document Object Model (DOM) based on XML (Extended Markup Language). Communication between configuration server 336 and the various components of network portal system 100 is provided via requests for entries or requests for modification of entries. In one embodiment, the data is exchanged using an XML-based protocol.
>
> Alterations to the DOM tree are carried out by transactions, which include inserting new nodes or new key-value pairs describing user preferences. This also covers such things as modification of individual entries or deletion of an entire subtree as the entries are no longer needed. Configuration server 336 also contains different mechanisms for querying the status of transactions or registering different listeners that are notified when alterations are made to configuration service elements and that in turn notify the applications affected.

Nowhere do these paragraphs of Hutsch teach or even suggest that "the plurality of configuration values in the hierarchical tree includes all of the configuration data values that are required by the at least one application component". These paragraphs merely teach that entries are stored in hierarchical form. This is not the same as the plurality of configuration values in the hierarchical tree including all of the configuration data values that are required by the at least one application component. Accordingly, this claim distinguishes over Hutsch.

With respect to claim 22, the Examiner states that Hutsch teaches "registering the at least one application component directly with the at least one inner node (see [0159] ad Fig. 16A)". Claim 22 now clearly recites "directly registering the at least one application component with the at least one inner node". As discussed above, Hutsch does not teach directly registering an application component with a node of any kind that includes configuration values indicating how that application component behaves with and/or interacts with other application components. Also, as discussed above, in Hutsch "listeners" only register with leaf nodes and are not directly registered with inners nodes. Accordingly, this claim and claim 23 distinguish over Hutsch.

With respect to claim 24, paragraph 29 of Hutsch has nothing to do with "at least one configuration value in the plurality of configuration values that is associated with a first application component overlaps with another configuration value in the plurality of configuration values that is associated with a second application component, wherein the at least one configuration value and the another configuration value are nested under a common sub-tree in the tree." Hutsch does not teach or suggest any type of nesting, let alone nesting configuration values when values of one application component overlap with values of another application component. Accordingly, this claim invention distinguishes over Hutsch.

Therefore, it is respectfully submitted that the rejection of claims 1, 2, 4-8, and 21-24 under 35 U.S.C. § 102(b) should be withdrawn.

Applicants are not conceding in this application that the claims that have been canceled are

not patentable over any cited reference. The claim amendments and cancellations have only been done to facilitate expeditious prosecution of allowable subject matter. Applicants respectfully reserve the right to pursue the canceled and other claims in one or more continuation and/or divisional patent applications.

No amendment made was related to the statutory requirements of patentability unless expressly stated herein. No amendment made was for the purpose of narrowing the scope of any claim, unless Applicants have argued herein that such amendment was made to distinguish over a particular reference or combination of references.

In view of the foregoing, it is respectfully submitted that the application and the claims are in condition for allowance or is at least in better form for appeal. Reexamination and reconsideration of the application, as amended, are requested.

If for any reason the Examiner finds the application other than in condition for allowance, the Examiner is invited to call the undersigned attorney at (561) 989-9811 should the Examiner believe a telephone interview would advance the prosecution of the application.

Respectfully submitted,

Date: September 2, 2008

By: /Stephen Bongini/
Stephen Bongini
Reg. No. 40,917
Attorney for Applicants

FLEIT GIBBONS GUTMAN
BONGINI & BIANCO P.L.
One Boca Commerce Center
551 Northwest 77th Street, Suite 111
Boca Raton, Florida 33487
Telephone: (561) 989-9811
Facsimile: (561) 989-9812